Session-Based Hotel Recommendations Dataset

As part of the ACM Recommender System Challenge 2019

JENS ADAMCZAK, trivago N.V., Germany
YASHAR DELDJOO, Polytechnic University of Bari, Italy
FARSHAD BAKHSHANDEGAN MOGHADDAM, Karlsruhe Institute of Technology, Germany
PETER KNEES, TU Wien, Austria
GERARD-PAUL LEYSON, trivago N.V., Germany
PHILIPP MONREAL, trivago N.V., Germany

In the year 2019, the Recommender Systems Challenge [17] deals for the first time with a real-world task from the area of e-tourism, namely the recommendation of hotels in booking sessions. In this context, we present the release of a new dataset which we believe is vitally important for recommendation systems research in the area of hotel search, from both academic and industry perspectives. In this paper, we describe the qualitative characteristics of the dataset and present the comparison of several baseline algorithms trained on the data.

ACM Reference Format:
Jens Adamczak, Yashar Deldjoo, Farshad Bakhshandegan Moghaddam, Peter Knees, Gerard-Paul Leyson, and Philipp Monreal. 2020. Session-Based Hotel Recommendations Dataset: As part of the ACM Recommender System Challenge 2019. *ACM Trans. Intell. Syst. Technol.* 1, 1, Article 1 (July 2020), 21 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

The rapid development of information and communication technologies and the web have transformed the tourism and travel domain. Today, travelers no longer rely on travel agencies but search for information themselves and compose their trips according to their specific preferences. Users have to choose from a multitude of options and recommender systems for travel and tourism can be practical tools to overcome the inevitable information overload. Such developments in e-tourism have been studied at the RecSys Conference[1] and the RecSys Workshop on Recommenders in Tourism (RecTour)[2] series, for example.

Recommending hotels and other travel-related items is still a difficult task as travel and tourism is a very complex domain. Planning a trip usually involves searching for a set or package of products that are interconnected (e.g., means of transportation, lodging, attractions), with rather limited availability, and where contextual aspects may have a major impact (e.g., time, location, social

[1]https://recsys.acm.org
[2]http://www.ec.tuwien.ac.at/rectour2019/

Authors' addresses: Jens Adamczak, jens.adamczak@trivago.com, trivago N.V., Düsseldorf, Germany; Yashar Deldjoo, yashar.deldjoo@poliba.it, Polytechnic University of Bari, Bari, Italy; Farshad Bakhshandegan Moghaddam, farshad.moghaddam@student.kit.edu, Karlsruhe Institute of Technology, Karlsruhe, Germany; Peter Knees, peter.knees@tuwien.ac.at, TU Wien, Vienna, Austria; Gerard-Paul Leyson, gerard-paul.leyson@trivago.com, trivago N.V., Düsseldorf, Germany; Philipp Monreal, philipp.monreal@trivago.com, trivago N.V., Düsseldorf, Germany.

2157-6904/2020/7-ART1 $15.00
https://doi.org/10.1145/1122445.1122456

context). Users book much fewer hotels than, for example, listen to music tracks, and, given the financial obligation of booking a stay at a hotel, users usually exhibit a strong price sensitivity and a bigger need to be convinced by any given offer. Besides, travelers are often emotionally connected to the products and the experience they provide. Therefore, decision making is not only based on rational and objective criteria. As such, providing the right information to visitors to a travel site, such as a hotel booking service, at the right time is challenging. Information about items such as hotels is often available as item metadata. However, usually, in this domain information about users and their goals and preferences is harder to obtain. Systems need to analyze session-based data of anonymous or first-time users to adapt the search results and anticipate the hotels the users may be interested in.

Recommendation systems for hotels and accommodations exist in different forms and scopes of application [3]. In this paper, we introduce a new public dataset of real-world hotel search sessions released by trivago.[3] We analyze the dataset and present descriptive statistics to highlight the information contained. We additionally compare how several baseline algorithms perform when trained on the data. This paper focuses on the description of the dataset used in the RecSys Challenge 2019. The RecSys Challenge is an annual competition that is held in conjunction with the ACM Conference for Recommender Systems. In each challenge, an industrial partner presents a dataset as well as a problem statement to the public. To get a more intuitive understanding of this particular example of the applicability of the data, we will briefly introduce the task posed to participants of the 2019 challenge.

The dataset presented here is tracking interactions of users with the trivago website. Users can search for accommodations, destinations, or points of interest (POIs), and get recommendations presented in the form of a list of results. The users have the option to interact with accommodations shown in the result list by clicking on the content of an accommodation or by making a click that forwards to a booking site (i.e. a clickout). Users can leave the trivago website, and return at a later point in time resulting in a new session. The task of the challenge is to use all the information about the behavioral, time-dependent patterns of the users and the content of the displayed accommodations to develop models that allow to predict which accommodations a user is most likely to click on when presented with a list of potential options. More details about what kind of information is available in the data can be found below in Section 3.

2 EXISTING DATASETS

Over of the last years, multiple datasets have been collected to aid the development of recommender system solutions for practical applications. However, domain-specific datasets that are relevant for the travel industry typically focus on descriptive content of information that can be scraped from a website. Other datasets that focus on user responses in recommendation settings, such as datasets from previous RecSys challenges, provide more information about website usage patterns but have so far not allowed drawing conclusions about the behavior of users in a travel-focused environment. With this paper and the associated dataset, we aim to bridge the gap between the different types of information that are currently available and provide relevant information for the development of recommendation systems in the online travel domain.

This section describes datasets from both scenarios, i.e. data from the travel domain, as well as the datasets that capture the user response in recommendation settings as presented in the previous RecSys challenges. These datasets are compared to the one presented in this paper from a structural perspective. A brief overview of the different datasets is shown in Table 1.

[3]https://www.trivago.com

Table 1. Overview of comparable datasets created for the development of recommender systems, as well as the dataset presented in this paper (shown below in bold text).

Dataset	Source	Row type	Description	Row count
BCOM19	Booking.com	review	reviews, scores, metadata (tag)	515K
DF19	Datafiniti	hotel	reviews, rating, and metadata	35K
GB19	goibibo.com	review	reviewsand rating	4K
MMTRIP19	MakeMyTrip.com	hotel	reviews and rating	20K
TRIP09	TripAdvisor	hotel	reviews and rating	235K
RS15	Yoochoose	session	clicks and purchases	33M
RS16	Xing	user	impressions and interactions	1B+8.8M
RS17	Xing	user	impressions and interactions	314M+7M
RS18	Spotify	user	playlists	1M
RS19	**trivago**	**session action**	**various types of user actions**	**19.7M**

2.1 Domain-specific datasets

Most of the existing datasets in the travel domain pertain to reviews or ratings of different hotels. Some of these datasets are listed below. These datasets differ from the dataset presented in this paper in the sense that they do not contain any information about actual user interactions on travel websites. Rather, they describe metadata and content of hotel inventory and location information. We aim to complement this static information with a more dynamic description of interaction patterns that illustrate how users are responding to content that is provided for them.

BCOM19 [5]: This dataset consists of 515K customer reviews and scores of 1,493 luxury hotels across Europe. It also supplies fine-grained information about reviews/reviewers, e.g., if the reviews are positive/negative, the total number of reviews, the nationality of reviewers along with tags assigned by reviewers to hotels, and hotel location (longitude and latitude). The data was scraped from Booking.com.

DF19 [10]: This dataset provides a list of 1,000 hotels and 35K reviews provided by Datafiniti's Business Database. The dataset supplies additional metadata such as hotel location, hotel name, rating score, review data, title, username, and so on.

GB19 [12]: This dataset supplies a subset, i.e., 4K, of a bigger dataset extracted from goibibo.com, a leading travel portal in India. It contains a wide range of metadata, such as information specific to hotels (category, description, facilities, star rating, image count), location, point of interest, and others. The original dataset contains information on 33K hotels.

MMTRIP19 [21]: Similar to *GB19* [12], this dataset is a smaller version of a large dataset taken from MakeMyTrip.com, a major travel portal in India. This dataset also includes metadata information (hotel overview, star rating, image rating), location, and rating. The original dataset contains information on 615K hotels and is available on DataStock, a data repository website supplying a historical record of several industries.

TRIP09 [28]: This dataset contains hotel reviews from TripAdvisor in a month (from February 14, 2009, to March 15, 2009). Besides the overall rating, this dataset contains additional information

such as aspect ratings in each review: value, room, location, cleanliness, check-in/front desk, service, and business service ranging from 1 star to 5 stars.

2.2 Previous RecSys Challenge datasets

Since the currently publicly available datasets from the travel domain lack information about user responses, they can not be used to build and evaluate recommendation algorithms. To make the reader aware of more similar datasets that have been successfully used in the past to evaluate novel recommendation system approaches, we offer a comparison with datasets from past RecSys challenges. The last four RecSys challenges are described below.

RS15 [4, 32]: This dataset comprises of session-based clicks and purchases of users using an e-commerce website. The task given by the industry partner is to predict whether a user is going to buy an item or not. If a purchase is predicted, a prediction of which item the user buys is also required. This dataset was provided by Yoochose.

RS16 [1, 30]: This dataset contains user-based impressions and interactions using a job posting website (XING). Metadata about the job postings are also provided. The task for this data set is to predict which job postings are likely to be relevant to the users. This dataset was provided by Xing.

RS17 [2, 31]: This dataset is similar to RS16, but both the dataset and task are modified to include a commercialization dimension. Participants are asked to balance relevance and revenue, as well as tackle the novelty/sparsity of information about new job postings. This dataset was also provided by Xing.

RS18 [6, 14]: This dataset contains user-based playlists by users of a music streaming website. Metadata about the tracks are also provided. The task is to generate an automatic playlist continuation based on the users' existing playlists [25]. This dataset was provided by Spotify.

The dataset presented in this paper (*RS19*) consists of anonymous session actions of trivago users and was opened to the public in the RecSys Challenge 2019 competition [17]. These actions not only included impressed/clicked hotels but also other website interactions, such as filter usage or search types. This introduces a different dimension to the existing datasets in the travel domain, which tend to be more centered around accommodations and their properties. In comparison to the previous RecSys challenges, our dataset introduces a finer granularity of user interactions with a website.

3 DATASET DESCRIPTION

This section summarizes the characteristics and data structure of the data and highlights some interesting features of it.

3.1 Overview

The dataset consists of the sequential website interactions of users visiting the trivago website between November 1st, 2018 and November 8th, 2018, as shown schematically in Figure 1. The data contains a wide variety of user interactions which include, for instance, making a click that forwards to a booking site (i.e. a clickout), an interaction with an item image, or a selection of a filter. An overview of all interaction types that are contained in the data can be found in Table 2.

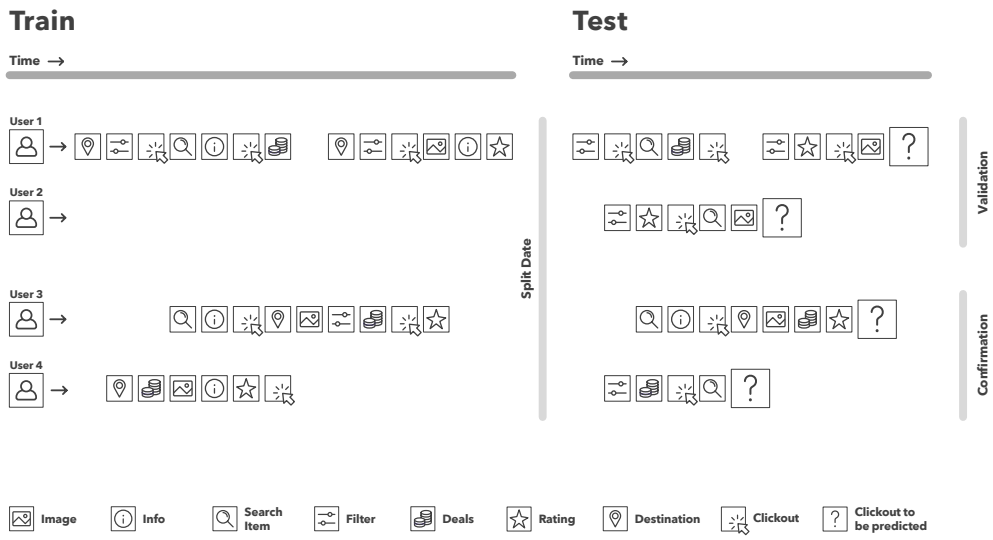

Fig. 1. Schematic illustration of the dataset and how it is split between train and test set. Each icon in the schematic represents a different type of website interaction, such as clicking on item content (image, info, rating, deals), refining the search parameters via filtering, or triggering new searches for accommodations (items) or destinations. All interactions that can be performed by the users and that are indicated by the icons in the schematic are in more detail described in Table 2. Gaps between consecutive interactions indicate the start of a new user session. The train set contains sessions before November 7th, 2018, while the test set contains sessions after said date. The item_id of the final clickout (shown as the box with the question marks) have been withheld. Note that the question mark refers only to the accommodation identifier that needs to be predicted and not the action type and that every event for which a prediction needs to be made is a clickout. For the evaluation of the leaderboard, the test set has been split into a confirmation and a validation set on a user basis.

Table 2. Description of action types and reference values for each possible session interactions.

Action type	Reference	Description
clickout item	item id	Click on item and forwarding to a booking site
interaction item rating	item id	Interaction with a rating or a review of an item
interaction item info	item id	Interaction with item information
interaction item image	item id	Interaction with an image
interaction item deals	item id	Click on the *View more deals* button
change of sort order	sort order name	User changes the sorting order
filter selection	filter description	Filter selection, e.g. four stars
search for item	item id	Search for the name of an accommodation
search for destination	city name	City search, e.g. Austin, TX
search for POI	POI name	Point of interest (POI) search

Table 3. General statistics of the RecSys Challenge 2019 dataset.

	training dataset	test dataset	item metadata
file size (MB)	2,100.81	534.91	257.90
row count	15,932,992	3,782,335	927,142
column count	11	11	2
unique sessions	910,683	291,381	-
unique users	730,803	250,852	-
clickouts	1,586,586	528,779	-
min. date	2018-11-01	2018-11-08	-
max. date	2018-11-07	2018-11-09	-
avg. sessions per user	1.246	1.162	-
avg. actions per user	21.802	15.078	-

Each website interaction corresponds to a specific timestamp in the dataset. Multiple website interactions can appear in a user session. A session is defined as all interactions of a user on a specific trivago country platform with no gaps between the interaction timestamps of > 60 minutes. If a user stops interacting with the website and returns after a couple of minutes to continue the search, the continued interactions will still be counted to belong to the same session. Because of the grouping of website interactions into sessions, the interactions are in the following often referred to as session actions. For each session interaction, data about the context of the interaction is provided, e.g. the country platform on which the interaction took place or the list of items that were shown at the moment of a click and the prices of the accommodations. Metadata for each of the accommodations are provided in a separate file.

3.2 File descriptions

The dataset used in the challenge consists of 3 files; train.csv, test.csv, and item_metadata.csv. General statistics of these files are summarized in Table 3. The first two files contain the session actions of various uses, while the last file contains information about item amenities that can be cross-referenced for additional information.

In addition to these 3 files that were made public to the participants in the course of the challenge, we make available the ground truth data used for evaluating the submissions. The ground truth data consists of a validation.csv file used for the calculation of the public leaderboard, and the confirmation.csv file used for the calculation of the final leaderboard. Both files are a subset of the test_ground_truth.csv file that contains the information of accommodation that has been clicked. The file can be accessed from a dedicated website[4].

The following data set descriptions focus on the 3 public datasets.

3.2.1 Session actions files. Each row in the session action files (train.csv and test.csv) corresponds to a particular user action in a given session. The schema of these files is shown in Table 4. The split between the train and test sets were done at a particular split date. That is, sessions that occurred before November 7th, 2018, were put into the train set, while those that occurred after were put into the test set. The target of the test set are items clicked out at the end of the sessions in the test set.

A user can perform a wide range of actions to interact with the item list. Depending on the action type, the 'reference' column could represent different things, such as an item identifier or a

[4]https://recsys2019data.trivago.com/

Table 4. Column descriptions of the session action files (train.csv and test.csv).

Column name	Data type	Description
user_id	String	Identifier of the user
session_id	String	Identifier of the session
timestamp	Timestamp	UNIX timestamp for the time of the interaction
step	Integer	Step in the sequence of actions within the session
action_type	String	Description of the user action at this session step
reference	Integer	Reference value for the different action types
platform	String	Country platform that was used for the search
city	String	Name of the destination of the search context
device	String	Device that was used for the search
current_filters	String	Pipe-separated list of filters that were active at the timestamp
impressions	String	Pipe-separated list of items that were displayed to the user
prices	String	Pipe-separated list of prices of impressed items in Euro

city name. For instance, if a user interacts with the rating of an item, the reference value contains the identifier of the item that was interacted with. A list of those actions is summarized in Table 2.

In addition to the user actions, the train.csv and test.csv files also contain information about the accommodations that were displayed to the user at the time a clickout was made. An accommodation that is displayed is referred to as being 'impressed' and all displayed accommodations are stored in the 'impressions' column. Each row in that column is a list of accommodations (items) in the order in which they were displayed on the website. In case the user action was not a clickout, the impression column is left empty.

To illustrate the different session actions more concretely, Table 5 shows an example of actions performed in a session by a user from the US platform has used trivago on a desktop device. The actions in this session are the following:

(1) User searches for Barcelona, Spain (action type: search for destination, reference: Barcelona, Spain).
(2) The 'focus on distance' filter is activated. At this point, the current_filters column indicates that this is the only filter that is active (action type: filter selection, reference: Focus on Distance).
(3) User searches for a point-of-interest (POI), the Port de Barcelona (action type: search for POI, reference: Port de Barcelona).
(4) User viewed at the 'More Deals' button on item 40255. The 'focus on distance' filter is no longer activated (action type: interaction item deals, reference: 40255).
(5) The user clicks out on item 40225. The full list of displayed items and their associated prices can be seen in the 'impressions' and 'price' columns (action type: clickout item, reference: 40225).
(6) User searches for item 81770 (action type: search for item, reference: 81770).
(7) User interacts with the item information of item 81770 (action type: interaction item info, reference: 81770).
(8) User clicks out on item 81770. The full list of items and their associated prices can be seen in the 'impressions' and 'price' columns (action type: clickout item, reference: 81770).

The dataset contains detailed information about session interactions and at the same time exhibits sparsity and imbalances in certain dimensions that make it challenging to derive personalized and

Table 5. A Sample Row of the Session Actions file (train.csv and test.csv).

step	action_type	reference	current_filters	impressions	prices
1	search for destination	Barcelona, Spain	-	-	-
2	filter selection	Focus on Distance	Focus on Distance	-	-
3	search for POI	Port de Barcelona	Focus on Distance	-	-
4	interaction item deals	40255	-	-	-
5	clickout item	40255	-	6744|40181|...	162|91|...
6	search for item	81770	-	-	-
7	interaction item info	81770	-	-	-
8	clickout item	81770	-	6832|40396|...	347|245|...

Table 6. Column description of the item metadata file (item_metadata.csv).

Column name	Data type	Description
item_id	Integer	Identifier of the accommodation. Used in the reference values for item related action types.
properties	String	Pipe-separated list of properties of items

context-specific recommendations based on previous interactions. About a third of the users (33.6%) did not interact with any items before making the final clickout. The top 10 locales constitute 62.3% of the total actions. The majority of actions (74%) are interactions with the item image.

3.2.2 Item metadata. Apart from the session action files, a separate file containing the item metadata is provided in the dataset ('item_metadata.csv'). The column descriptions of this file are described in Table 6. Item properties can be the classification of the accommodation (e.g. the hotel star rating, bed-and-breakfast or 'serviced apartment'), amenities of the accommodation (e.g. balcony, free wifi, swimming pool) or available services (e.g. laundry service, massage). There are 157 unique properties in the dataset.

This dataset can enrich the previous datasets for actions relating to items or lists of items by referencing these items using the item_id. There is a large imbalance in the number of properties per item. The mean number of properties is 19.7, but 20% of the items have only 1 property.

3.3 Dataset characteristics

Before exploring the structure of the dataset in the context of building predictive models in Section 4, we discuss some of the characteristic properties of the data. The data is a collection of user interactions across 55 country platforms. It allows us to inspect individual user preferences in terms of selected destinations and accommodations, and the analysis of website usage patterns on platform level.

3.3.1 User search preferences. A direct way for a user to specify a preference is to search on the website for a destination, a POI, or an accommodation. This information is contained in the dataset in the form of search-related action types. The most commonly searched destinations and POIs are displayed in Table 7. User preferences specified in this way are useful in the development of recommender systems. In reality, information about these preferences is often sparse. This is also true to a certain extent to the dataset presented here. The number of searches for the individual

Table 7. Top 10 searched destinations and points of interest (POI) in the training data set. The preferences are given in the absolute number of searches and in the percentage of searches that a particular city or POI received relative to the total number of city or POI searches.

Rank	City	Searches	[%]	POI	Searches	[%]
1	London, United Kingdom	6,421	1.59	Tokyo Disneyland	2,093	1.52
2	Paris, France	4,858	1.21	Kyoto Station	1,303	0.95
3	New York, USA	4,667	1.16	Las Vegas Strip	1,238	0.90
4	Amsterdam, Netherlands	3,609	0.90	Tokyo Station	1,233	0.90
5	Las Vegas, USA	2,960	0.73	Shinjuku Station	1,114	0.81
6	Berlin, Germany	2,764	0.69	Melbourne CBD	1,099	0.80
7	Barcelona, Spain	2,686	0.67	Times Square	1,082	0.79
8	Rome, Italy	2,615	0.65	Sydney CBD	974	0.71
9	Cancun, Mexico	2,460	0.61	Osaka Station	960	0.70
10	São Paulo, Brazil	2,357	0.58	Hakata Station	927	0.67

Table 8. Top 10 trivago country platforms in terms of total interactions that users from that platform had with the website in the training data set. The interactions in the top 10 are relatively evenly distributed across the different platforms with Brazil and the USA standing out with a higher share of overall interactions.

Platform	Country	Interactions	[%]
BR	Brazil	2,634,304	16.53
US	USA	1,627,520	10.21
DE	Germany	1,001,105	6.28
UK	United Kingdom	918,900	5.77
MX	Mexico	833,785	5.23
IN	India	679,747	4.27
AU	Australia	595,003	3.73
TR	Turkey	564,271	3.54
JP	Japan	547,480	3.44
IT	Italy	527,046	3.31

destinations and POIs is small compared to the large number of total interactions that are registered for the top platforms, as can be seen in Table 8. Searches for destinations and POIs combined make up for only 3.4% of all interactions. The search for an item that would be very valuable in the context of the challenge as an explicit signal of interest in a particular accommodation, accounts for only 0.96% of all interactions. Directly specified user preferences alone are not sufficient to reliably predict clicked accommodations. The explicit search information needs to be complemented by the other, implicit, session actions that capture the website usage patterns and that are more frequently performed.

3.3.2 Website usage patterns. Usage patterns can be described by the temporal dependency of user interactions. Figure 2 displays the number of interactions per world region for the training and test time frame of the dataset. The world regions group the different trivago platforms by similar time zones according to geographic regions in the United Nations M49 standard. The regions are reduced to three geographic areas, America (North America, and Latin America and the Caribbean), Asia

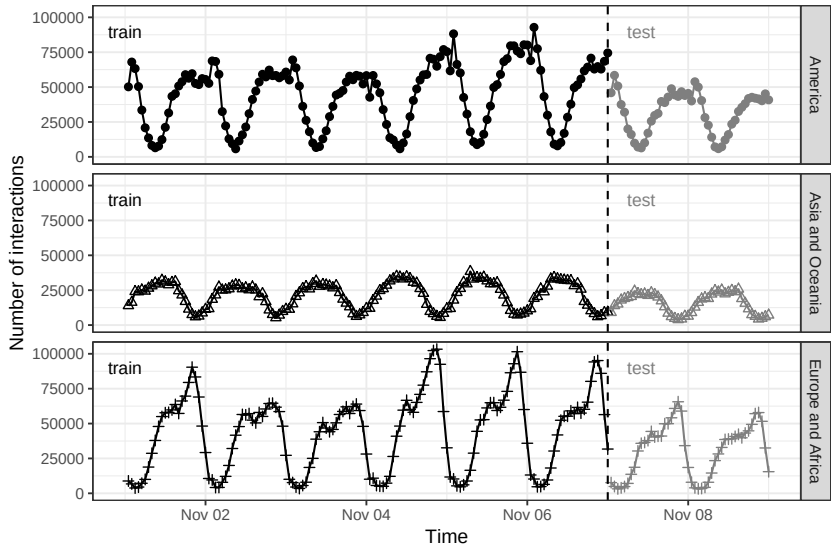

Fig. 2. Temporal patterns of user interactions for the different world regions. The Time axis displays the date and time in Universal Time Coordinated (UTC). Interactions happen in a pattern of daily seasonality that has different peaks depending on the current timezone and daily browsing preferences.

and Oceania, and Europe and Africa. There are clear hourly variations due to the normal seasonality of common human daily routines, i.e. less usage of the website late at night for the respective time zones. Apart from the daily seasonality, there are differences between the different days of the week. The time series for Europe and Africa shows a more complex profile that potentially can be decomposed into more specific user patterns for the individual platforms that constitute this world region. There seem to be geographical differences between the regions.

The frequency of the type of interactions that are performed is shown in Table 9. The most common website interaction is the interaction with the image of an accommodation on the result list page. It accounts for 74% of all interactions. The next most common interaction is the clickout on the item with almost 10% of the total number of interactions. In addition to their overall frequency in

Table 9. Top 10 trivago country platforms in terms of interactions in the training data set.

Action type	Occurrence	[%]
interaction item image	11,860,750	74.44
clickout item	1,586,586	9.96
filter selection	695,917	4.37
search for destination	403,066	2.53
change of sort order	400,584	2.51
interaction item info	285,402	1.79
interaction item rating	217,246	1.36
interaction item deals	193,794	1.22
search for item	152,203	0.96
search for POI	137,444	0.86

the dataset, interactions with an image of an item are useful indicators to infer what accommodation is eventually clicked as they are often directly followed by a clickout to the same item. Figure 3 shows the frequency of action types that are the direct precursor of a clickout event. The direct precursor action is the action that happened at the step directly before a clickout in a session of a user. The most common event that happens directly before a clickout is another clickout. This is the case for about 30% of all clickouts. In 21% of all cases, the clickout before happened on another item than the item that was eventually clicked. In 9% the clickout goes to the same item. The action types that are the next likely to happen right before a clickout is the interaction with the image of an item. This is not surprising as the image interaction is the most common action type overall. It is still interesting to see that for the interaction with an image of the item it is more likely that the eventual clickout goes to the same item than for the clickout action type. The item interaction appears to be a strong indicator of the final clickout choice if it happens directly before the clickout. It is furthermore remarkable that in almost 20% of cases there is not direct action type preceding a clickout, i.e. the clickout is the first action type that is recorded in the session. This makes the development of models based on previous session information challenging and requires the pooling of additional information across sessions and users.

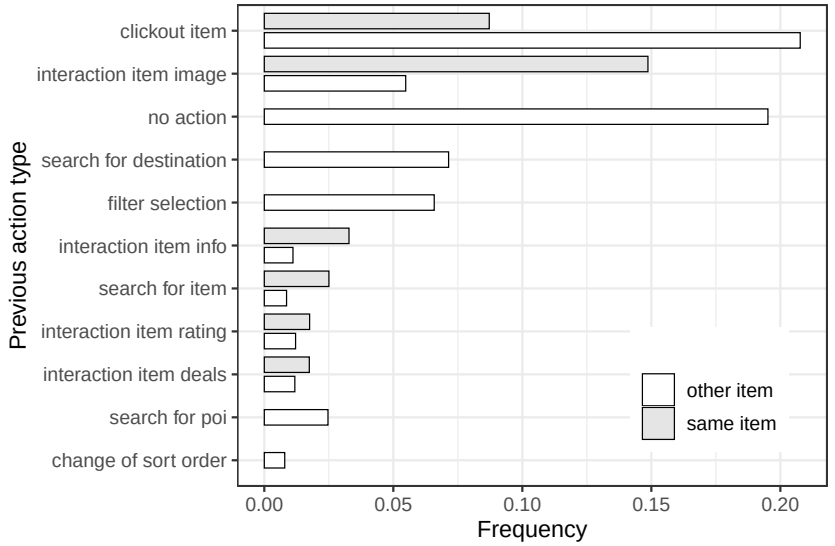

Fig. 3. Overview of the frequency of the action types that directly precede a clickout. The most common event that happens before a clickout is another clickout. Interactions with the image of an item are disproportionately often followed by a clickout that goes to the same item instead of going to another item.

Another difficulty is that the amount of data on interactions and clickouts is rich in aggregation but very unevenly distributed, leading to sparsity on session and user level. Figure 4 illustrates the distribution of data using the example of interactions per session in the training set. The majority of sessions consist of only one interaction. Few sessions are available that have complex user patterns and more than 100 interactions. The sparsity of the data is even more extreme for information about individual users. Out of the 730,803 users in the training set, 84% have only one session to account for. To infer the identity of a clicked item, additional information about individual users and their respective browsing history inevitably needs to be complemented by information about the context of the clickout. In that regard, the dataset contains information about over 36 million prices of

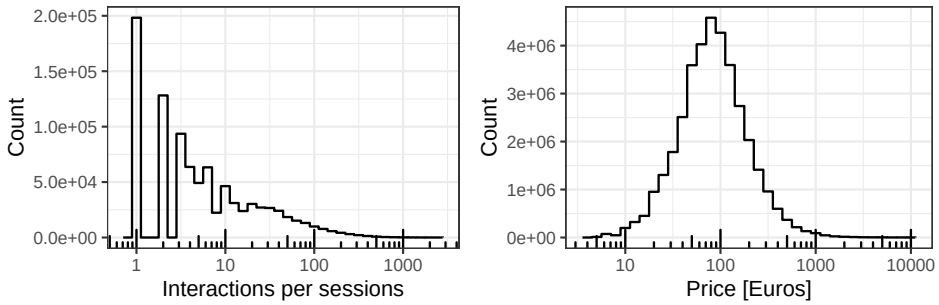

Fig. 4. Distribution of data for the number of interactions per session (left panel) and prices of accommodations in the impression list (right panel) in the training set. Note that the data is presented on a log scale, illustrating the uneven distribution of data. The majority of sessions consists of only one interaction.

accommodations that were displayed to the users in the impression lists. A histogram of these prices is shown in the right panel of Figure 4. In contrast to the very sparse session interaction and user history information, prices are distributed closer to a log-normal distribution, revealing the range of the most commonly displayed prices when presenting the data on the logarithmic scale.

In the next section, we will explore methods that use the information in the data to infer the accommodation of the clickouts that were withheld in the test data.

4 EXPERIMENTS

The dataset can be utilized for a variety of applications. We restrict our exemplary exploration to the context of the 2019 RecSys Challenge problem formulation. The challenge is essentially a ranking task. Participants have to take the list of displayed accommodations in the impression list and submit a list of items in a new order according to the item click propensity for each missing clickout. In this section, we present a set of baseline algorithms that illustrate how to train models on the data and calculate predictions. The selected models draw inspiration from the context of the challenge and highlight specific aspects of the dataset.

Sophisticated algorithms exist that aim at exploiting the session-based nature of data to predict user preferences and provide recommendations, e.g. GRU4Rec [13]. We refrain from going into more detail about these kinds of algorithms in the experimentation section for two reasons. First, it has been shown that for many datasets more basic algorithms can achieve competitive performances that are easier to implement and have more moderate requirements on training time and faster inference speed [19]. This is in line with the goal that algorithms, presented here can be run within a short time in a local environment and do not have overly challenging hardware requirements.[5] Secondly, the dataset presented in this paper is peculiar in the way that it inhibits an extreme sparsity of data for individual sessions and users and often short session lengths. This makes it challenging to apply known session-based recommendation systems, and among the top-performing participants of the challenge indeed no team did. Participants rather opted to extract meaningful session-based features and revised methods that drew information across different sessions and users.

In the experimental part of this paper, we will focus on benchmarks that describe certain key aspects of the data that turned out to be important to achieve high performance in the challenge. We

[5] All models presented have been run on a 2.7 GHz Intel Core i5 machine with 16 GB 1867 MHz DDR3 ram within the time frame of a few minutes.

will contrast the benchmarks with the more advanced state-of-the-art models that were developed in the course of the challenge in Section 4.7. The code to reproduce the experiments with the dataset of the challenge is stored in a publicly available repository[6].

4.1 Evaluation

We evaluate all presented algorithms with the metric used in the challenge, the mean reciprocal rank (MRR). For a list of ranked items, the reciprocal rank is the multiplicative inverse of the rank of the first positive response. For example, if an item i is clicked or booked on position rank_i in the result list (counting from the top), the reciprocal rank is denoted as $1/\text{rank}_i$. The mean reciprocal rank is the average of all reciprocal ranks for a given number N of inspected results lists,

$$MRR = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{\text{rank}_i} \tag{1}$$

In addition to the MRR metric, we provide a second performance measure in the form of the average precision@3. Precision is defined as the fraction of relevant items, in this case, clicked, to the user for a given search query. In this concrete example, we measure if the clicked item I_c appeared in the top 3 submitted results. We annotate this by the indicator function $[I_c \in top_3]$. The function is 1 if the item appears in the top results and 0 if it does not. The average precision@3 is the average across all clicks N that have to be predicted.

$$Avg.Precision@3 = \frac{1}{N} \sum_{i=1}^{N} \frac{[I_c \in top_3]}{3} \tag{2}$$

To evaluate the algorithms the test set is split on a user basis into a validation set and a confirmation set. The performance achieved on the validation set corresponds to the public leaderboard that was available for participants during the challenge. The performance achieved on the confirmation set corresponds to the final performance as evaluated after the end of the challenge.

4.2 Basic benchmarks

In the following sections, we describe methods build upon the information contained in the training data set only. The algorithms are afterward applied to the test data to rank the list of impressed items for each missing click based on the relevance of the items determined by each method.

For each clickout, the dataset contains information about the impressed accommodations, i.e. the list of items that were displayed to the user at the time of the clickout. The order of the accommodations in the impression list corresponds to the order in which they were displayed. In other words, the order corresponds to the original sorting of the list as calculated by a baseline algorithm that was used in the production of the dataset. We can make use of this order to measure the quality of this baseline algorithm by not re-ranking the items in the impression list and leaving the order unaltered for the evaluation. We refer to this order as *Position* in Table 10 indicating that the position in which the items appeared originally is responsible for the performance on the evaluation sets. Essentially, this means ignoring any session-based data. The position of the hotel in the original impression list was regularly identified as a top feature in the models of the challenge participants. This benchmark gives an indication of what value for the final prediction this feature has and what results could have been achieved if a submission had relied solely on this feature.

[6]https://github.com/trivago/recsys-challenge-2019-benchmarks

As a second baseline, we introduce the result that can be achieved when submitting a randomly shuffled list. This is not per se a ranking algorithm. The score will be influenced by the typical list length of impressed items and can be regarded as a lower bound that each algorithm should beat. This benchmark is referred to as *Random* in Table 10 and as expected has the lowest MRR and average precision value of all algorithms presented. This benchmark serves as a lower bar for all evaluated options and reveals that there is variability in the length of the impression lists provided in the data set. The maximum number of items that appear on the list is 25. If all lists were of the same length, we would expect an MRR of 0.15, i.e. the mean of the reciprocal ranks from 1 to 25. There must be significant instances of lists in the dataset that are shorter.

4.3 Popularity based methods

Popularity based recommendation algorithms measure the importance of individual items based on how frequently users interacted with them in the past. They cannot account for changes in user interest and are limited in their capability to explore new inventory but they usually provide a reasonable benchmark for other algorithms as they well capture recent user interest.

4.3.1 Absolute item clicks. We present two variants of popularity based algorithms. The first one defines popularity of item i as the absolute number of clicks the item received in the training phase. For the final predictions on the test set, items will be sorted in decreasing number of training clicks. If multiple items in an impression list have not received any clicks in the training phase they will be ordered according to their original position. The performance of the popularity based result can be seen in Table 10 under the name *Popularity-absolute*. The score achieved is lower than the *Position* estimate indicating that the popularity based method can only provide a very basic benchmark in this scenario.

4.3.2 Distinct users. The second variant of a popularity based method characterizes popularity for item i by the number of distinct users that clicked on that item. Again the assessment is made for the training phase and the method is applied to the test set. The performance of this variant (*Popularity-users*) is slightly higher than *Popularity-absolute* as it reduces the influence of repeated clicks on the same items.

The low performance of the popularity based algorithms illustrates again the fact that the dataset is very sparse. The majority of the predictions that need to be made are for first-time users looking for items with little historic information.

4.4 Nearest Neighbor algorithms

Nearest neighbor algorithms calculate predictions by defining a similarity measure between items and for a target item recommend the n closest items in the given similarity measure. They have been shown to deliver satisfactory results for various use cases [26]. We provide results for two versions of nearest neighbor approaches. For both versions, we identify the last item that a user interacts with in a given session as the target item and calculate the similarities between this item and all items in the impression list. The two methods differ in the way they calculate the similarity measure.

4.4.1 Item based. For the item based nearest neighbor approach, we calculate the similarity based on the item metadata. For this purpose, we construct a vector for each item characterizing the metadata that is available for that item in the item_metadata.csv. More concretely, the vector for each item is a binary vector with length 157, i.e. the number of distinct properties in the item metadata set. Each element e of the vector is set to 1 if an item has property e and is otherwise left to be 0. Once the metadata vectors have been constructed, we identify for each user the last item

that was clicked in the test set before the final list of impressed items was presented. Subsequently, we calculate the cosine similarity between the vector of the last clicked item and the vector of each item that appears in the impression list. We then sort the list in decreasing order of similarity, i.e. showing items on top that are more similar to the last clicked item. The item metadata based method is indicated as *nn-Item* and performs better than all previous benchmarks.

4.4.2 Interaction based. The interaction-based approach works similarly to the item metadata based one and uses the same method as described in [13]. A similarity between two items is measured based on the number of session co-occurrences of these items. Concretely, we first construct a binary vector for each item i in the length of the number of sessions in the training data. The value for item i and session s will be set to 1 in case the item appeared in the session, otherwise the value will be set to 0. For for an item to qualify as being present in a given session it needs to have had at least one user interaction in that session. Similar to the previous case we calculate the cosine similarity of the interaction vector of the last clicked item and each item in the impression list and sort the list accordingly.

The fact that the interaction-based nearest neighbor algorithm performs so much better than the item-based one illustrates that for the given dataset interaction data appears to be more useful than the content that is provided for the items. Both nearest neighbor models calculate similarities between the last interacted item and items in the impression list. Especially the interaction-based method (*nn-Interaction* in Table 10) performs surprisingly well and shows that previous session interactions are a strong signal to predict how likely a user is to click on a particular item. The importance of incorporating recent interaction data as a feature into the predictions for a particular user was recognized by the challenge participants and heavily used in the feature engineering process in the development of the models of the top-performing solutions.

4.5 Binary classification

Some of the participants of the challenge formalized the challenge task in the form of a binary classification problem or a click prediction task. In that definition, a click probability for each item in the impression list needs to be calculated based on different item and session related features. To tackle the challenge in this framework we first transform the training data to restrict it to the clickout action type and expand the impression and price list to have each row represent an impression. We add a column that indicates if a given item has been clicked or not as the target variable. To calculate the final predictions we transform the test data in the same way and apply the model that was trained on the training data. There are multiple options to solve a click prediction problem, including linear models, boosted tree methods, and also certain neural network architectures. A standard model that is used widely in the industry is the logistic regression model. It is known to perform especially well in cases where the signal to noise ratio is low and on smaller datasets [23]. Our dataset is not small but very sparse which makes it hard to detect the signal.

We select four input item-specific features, the position of the item in the original listing, the price of the item, the number of previous interactions that the user had with a given item, and an indicator if an impressed item is the last item the user interacted with. We train the model on the training set and apply it to the test set data. The items in the impression list for each missing click are sorted in decreasing click probability. With this basic feature set, the algorithm already performs very well as can be seen in table 10 for entry *Logistic Regression*.

The good performance of the logistic regression model hints at the importance of the selected features. The model itself is basic and the selection of relevant features makes it possible to achieve an acceptable result without any model optimization and parameter tuning.

4.6 Learning to Rank

In the previous example, the relevance of an item was calculated according to its probability of generating a clickout. Depending on the scenario, the absolute click probability of an individual item might be of less importance than the likelihood that more relevant items appear higher in the list than less relevant items. Techniques aimed at learning to rank address this problem by looking at combinations of items and introducing loss functions that quantify if combinations of items are ranked appropriately.

One particular example of a learning to rank model is LambdaRank. In this approach, a ranking is optimized by considering pairs of items. Model parameters are determined by weighting the gradient of a pairwise loss function by the change in ranking accuracy that occurs when swapping two items. Typically the normalized discounted cumulative gain (NDCG) is used to determine the ranking accuracy. NDCG is equivalent to the MRR in cases in which only a binary outcome is measured and is therefore a fitting optimization measure for our use case.

In out approach we use the LightGBM implementation of the LambdaRank method [16]. The loss function in this implementation consists of gradient boosted trees. We specifically pick LightGBM to calculate the benchmark because it was chosen by many participants of the challenge. Some of the top-performing solutions used this implementation with slightly different choices of the objective function and boosting method.

The input data for the model is prepared in the same way as for the logistic regression model. We use the same feature set as before as an input for the ranking model. Compared to the logistic regression model, LightGBM needs an additional input that specifies the search query, i.e. the individual result list, that an impression belongs to in order to identify suitable pairs of items to compare. The size of the search query is equivalent to the length of each impression list and we feed this length into the model.

As opposed to a click probability, the LightGBM model produces a relevance score for each impressed item. As for the previous examples, we sort the items by decreasing relevance before evaluating the results. Table 10 shows the result of this scoring process under the reference GBMrank. This method outperforms all other tested approaches. If the LightGBM model with the few features introduced here would have been used in the challenge, it would have ended up in the top 15% of submissions. This illustrates that a careful selection of features in combination with a strong recommendation model is a promising approach for the given data. Unsurprisingly the LightGBM model was the workhorse in the development of a lot of models in the competition.

The described benchmark models only pinpoint some of the characteristics of the dataset. To achieve a better performance, more sophisticated models were developed by the challenge participants.

4.7 Algorithm performance

We present the results of our benchmark models and contrast them with the results of the state-of-the-art models that have been developed by participants of the challenge.

Table 10 summarizes the results for all tested benchmark approaches. The motivation for choosing a particular benchmark model has been provided for the description of each benchmark. In summary, the presented algorithms were selected to demonstrate the utilization of different features of the data sets that turned out to be relevant in the challenge. They are by no means exhaustive and do not claim to fully explore the characteristics of the data. Individual refinements for each of the chosen methods can potentially yield higher scores, as can the combination of different models and the integration of a much larger feature set. For reference, the highest MRR scores that were

Table 10. Overview of performance of baseline algorithms in decreasing order of performance.

Model	MRR		Average Precision@3	
	Validation	Confirmation	Validation	Confirmation
GBMrank	0.647	0.645	0.230	0.229
Logistic Regression	0.642	0.640	0.228	0.227
nn-Interaction	0.634	0.632	0.224	0.223
nn-Item	0.503	0.500	0.182	0.181
Position	0.502	0.500	0.181	0.181
Popularity-users	0.290	0.290	0.103	0.103
Popularity-absolute	0.288	0.288	0.102	0.102
Random	0.177	0.177	0.051	0.051

achieved in the associated challenge were 0.689 for the validation set and 0.686 for the confirmation set as opposed to 0.647/0.645 for the benchmark models.

5 CONCLUSION

We presented a new data set in the context of a recommendation challenge in the online travel domain. We illustrated the characteristics of the data and highlighted some use cases that have already sparked the interest of challenge participants. Initial exploration of algorithmic approaches to extract information from the data showed that many methods can be applied but also demonstrates that certain aspects of the data such as sparsity and imbalance make it challenging to reach high performance on the evaluation metrics. In addition to the experiments presented here, we will highlight some of the learnings that became apparent in the inspection of multiple models that were developed from different participants of the RecSys Challenge 2019 and demonstrate how they effectively make use of the data.

In the bigger scheme of understanding what methodological approaches are appropriate to deal with the data and predict the items that were clicked, we will use the following section to describe certain aspects that proved successful in the challenge and could be recognized as key ingredients for high-performing results across a variety of teams and submissions, namely feature engineering, gradient boosted models, and ensembling of prediction results.

5.1 Feature engineering

By far the most successful strategy was to include the extensive engineering of targeted features into the model building process. The participants realized that with a robust set of models and a reasonable validation setup it was hard to overfit on the validation set. As a consequence, the teams built and evaluated a large set of features, many of which made it to the final models. As an example, the winning solution used 220 numerical features that encoded the context of the query and 30 categorical one-hot encoded features [15]. Other submissions in the top 5 had a similar, and sometimes even higher, amount of up to 518 different features [20, 27].

Typical features included item and user features, impression related features, such as absolute and relative prices and characteristics of items that were shown together in a list, as well as session related features. Instead of using sequence aware models that exploit the time-dependent information of session data directly, the participants mostly opted to build sequential features that capture previous interactions and similarity between displayed items and past interactions. Examples for these kinds of features are the time difference between the target click and the

previous clickout [15], the number of interactions with a target item in the current session [27], the position of the last interacted item in the impression list [18], or a feature aimed at mimicking the browsing patterns of a user by inferring the expected position of the users' viewport based on previous session interactions [9]. In addition to more conventional features derived from statistics of the dataset, in some cases more complex embedding features were trained to transform the information that is contained in a session into latent variables that could be fed into the prediction models [7, 20].

Most of the feature engineering was done in the context of gradient boosted models. However, features were also engineered for Neural Network models that in other areas provide successful results without the need for extensive crafting of features. For the given data, it was found that even the performance of Recurrent Neural Networks (RNN) could be improved if they were fed with additional pre-processed data [11].

5.2 Gradient boosted models

Almost all teams did not solely rely on the predictions of a single model but followed an ensemble strategy to tweak the performance of their submissions (see Section 5.3). Among the individual models that went into the ensemble, gradient boosted tree methods were preferred. Four, out of the five top-performing teams, strongly relied on the output of gradient boosted models. One reason that the teams relied on these models was that they do not require a careful normalization of the features and allowed for the usage of raw features, which played well with the extensive feature engineering and the inclusion of a large number of estimator that was identified as the key to achieve the best performance on the given dataset.

A popular model choice was the LightGBM implementation of the gradient boosting method that convinced due to the flexibility to handle both classification and ranking objectives, multiple boosting options, and the low memory usage that paid of when training models on the large dataset of the challenge [15, 18, 20, 29]. A strong performance was also achieved with XGBoost models [8] that outperformed Deep Learning models with Transformer and Factorization Machine architectures in direct comparison [18, 27].

5.3 Ensembling

Virtually none of the top teams relied on the performance of a single model for the final predictions. The winning solution consisted of a combination of 37 LightGBM models [15]. Other teams were combining a lower number of models and the exact ensembling strategies differed between teams and depended on the underlying models. A common pattern was the development of different models and the combination of the predictions derived from these models via a linear blend [15, 18, 27] or stacking [9, 28]. Especially for the stacking approach, it was found useful to also include some of the top-performing feature values next to the predictions of the individual models.

Not all teams developed a set of heterogeneous models. Also in the case of similar individual models, aggregating predictions from runs of models that varied in the underlying training data or model parameters proved to be successful. For example, several teams used cross-validation to split the data, calculate predictions on the different folds, and ensemble them for the outcome [19, 22]. Others employed a self-averaging method that combined model outcomes derived with different initialization parameters to reduce the variance in the predictions stemming from Recurrent Neural Networks [11].

Table 11. Overview of performance of selected submissions for the RecSys Challenge 2019. Algorithm results are sorted in decreasing order of performance.

Team	Authors	MRR	
		Validation	Confirmation
LogicAI [15]	Jankiewicz et al.	0.689	0.686
Layer 6 AI [27]	Volkovs et al.	0.688	0.685
Meituan Dianping [29]	Wang et al.	0.686	0.684
RosettaAI [18]	Kung-Hsiang et al.	0.682	0.680
TU Dortmund [20]	Ludewig et al.	0.684	0.679
Polytechnic Institute of Viseu [11]	Gama et al.	0.681	0.679
Politecnico di Milano [9]	Damico et al.	0.679	0.677
NVidia [24]	Rabhi et al.	0.673	0.671
KAIST [22]	Oh et al.	0.673	0.670

Table 11 displays the results for selected submissions that were presented at the workshop on the RecSys Challenge 2019 at the 13th ACM Conference on Recommender Systems in Copenhagen 2019[7].

We hope that in the future the data can contribute to developing even more and new methods or test already existing ones in a different context and that the exploration of the data will spark the interest of researchers in a variety of areas.

REFERENCES

[1] Fabian Abel, András A. Benczúr, Daniel Kohlsdorf, Martha Larson, and Róbert Pálovics. 2016. RecSys Challenge 2016: Job Recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, September 15-19, 2016*. 425–426. https://doi.org/10.1145/2959100.2959207

[2] Fabian Abel, Yashar Deldjoo, Mehdi Elahi, and Daniel Kohlsdorf. 2017. RecSys Challenge 2017: Offline and Online Evaluation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017*. 372–373. https://doi.org/10.1145/3109859.3109954

[3] Jens Adamczak, Gerard-Paul Leyson, Peter Knees, Yashar Deldjoo, Farshad Bakhshandegan Moghaddam, Julia Neidhardt, Wolfgang Wörndl, and Philipp Monreal. 2019. Session-Based Hotel Recommendations: Challenges and Future Directions. *arXiv preprint arXiv:1908.00071* (2019).

[4] David Ben-Shimon, Alexander Tsikinovsky, Michael Friedmann, Bracha Shapira, Lior Rokach, and Johannes Hoerle. 2015. RecSys Challenge 2015 and the YOOCHOOSE Dataset. In *Proceedings of the 9th ACM Conference on Recommender Systems, RecSys 2015, Vienna, Austria, September 16-20, 2015*. 357–358. https://dl.acm.org/citation.cfm?id=2798723

[5] Booking.com. 2019. 515K Hotel Reviews Data in Europe. https://www.kaggle.com/jiashenliu/515k-hotel-reviews-data-in-europe. [Online; accessed 19-July-2015].

[6] Ching-Wei Chen, Paul Lamere, Markus Schedl, and Hamed Zamani. 2018. Recsys challenge 2018: Automatic Music Playlist Continuation. In *Proceedings of the 12th ACM Conference on Recommender Systems, RecSys 2018, Vancouver, BC, Canada, October 2-7, 2018*. 527–528. https://doi.org/10.1145/3240323.3240342

[7] Li Chen, Guanliang Chen, and Feng Wang. 2015. Recommender systems based on user reviews: the state of the art. *User Model. User-Adapt. Interact.* 25, 2 (2015), 99–154. https://doi.org/10.1007/s11257-015-9155-5

[8] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. Association for Computing Machinery, New York, NY, USA, 785–794. https://doi.org/10.1145/2939672.2939785

[9] Edoardo D'Amico, Giovanni Gabbolini, Daniele Montesi, Matteo Moreschini, Federico Parroni, Federico Piccinini, Alberto Rossettini, Alessio Russo Introito, Cesare Bernardis, and Maurizio Ferrari Dacrema. 2019. Leveraging laziness, Browsing-Pattern Aware Stacked Models for Sequential Accommodation Learning to Rank. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359563

[7] https://recsys.acm.org/recsys19/challenge-workshop/

[10] Datafiniti's Business Database. 2019. Hotel Reviews. https://data.world/datafiniti/hotel-reviews. [Online; accessed 19-July-2015].

[11] Ricardo Gama and Hugo Fernandes. 2019. An Attentive RNN Model for Session-based and Context-aware Recommendations: A Solution to the RecSys Challenge 2019. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359757

[12] Goibibo.com. 2019. Indian Hotels on Goibibo. https://www.kaggle.com/PromptCloudHQ/hotels-on-goibibo. [Online; accessed 19-July-2015].

[13] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based Recommendations with Recurrent Neural Networks. *CoRR* abs/1511.06939 (2015).

[14] https://www.spotify.com. 2018. RecSys Challenge 2018. https://2018.recsyschallenge.com/index.html. [Online; accessed 28-Aug-2019].

[15] Pawel Jankiewicz, Liudmyla Kyrashchuk, Pawel Sienkowski, and Magdalena Wojcik. 2019. Boosting algorithms for a session-based, context-aware recommender system in an online travel domain. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359557

[16] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 3146–3154. http://papers.nips.cc/paper/6907-lightgbm-a-highly-efficient-gradient-boosting-decision-tree.pdf

[17] Peter Knees, Yashar Deldjoo, Farshad Bakhshandegan Moghaddam, Jens Adamczak, Gerard Paul Leyson, and Philipp Monreal. 2019. RecSys challenge 2019: session-based hotel recommendations. In *Proceedings of the 13th ACM Conference on Recommender Systems, RecSys 2019, Copenhagen, Denmark, September 16-20, 2019*. 570–571. https://doi.org/10.1145/3298689.3346974

[18] Huang Kung-Hsiang, Fu Yi-Fu, Lee Yi-Ting, Lee Tzong-Hann, Chan Yao-Chun, Lee Yi-Hui, and Shou-De Lin. 2019. A-HA: A Hybrid Approach for Hotel Recommendation. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359560

[19] Malte Ludewig and Dietmar Jannach. 2018. Evaluation of Session-based Recommendation Algorithms. *CoRR* abs/1803.09587 (2018). arXiv:1803.09587 http://arxiv.org/abs/1803.09587

[20] Malte Ludewig and Dietmar Jannach. 2019. Learning to Rank Hotels for Search and Recommendation from Session-based Interaction Logs and Meta Data. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359561

[21] MakeMyTrip.com. 2019. Indian Hotels on Makemytrip. https://www.kaggle.com/PromptCloudHQ/hotels-on-makemytrip. [Online; accessed 19-July-2015].

[22] Jaehoon Oh, Sangmook Kim, Se-Young Yun, Seungwoo Choi, and Mun Yong Yi. 2019. A Pipelined Hybrid Recommender System for Ranking the Items on the Display. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359565

[23] Claudia Perlich, Foster Provost, and Jeffrey S. Simonoff. 2003. Tree Induction vs. Logistic Regression: A Learning-Curve Analysis. *J. Mach. Learn. Res.* 4, null (Dec. 2003), 211âĂŞ255. https://doi.org/10.1162/153244304322972694

[24] Sara Rabhi, Wenbo Sun, Julio Perez, Mads Burgdorff Kristensen, Jiwei Liu, and Even Oldridge. 2019. Accelerating Recommender System Training 15x with RAPIDS. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359564

[25] Markus Schedl, Hamed Zamani, Ching-Wei Chen, Yashar Deldjoo, and Mehdi Elahi. 2018. Current challenges and visions in music recommender systems research. *IJMIR* 7, 2 (2018), 95–116. https://doi.org/10.1007/s13735-018-0154-2

[26] Koen Verstrepen and Bart Goethals. 2014. Unifying Nearest Neighbors Collaborative Filtering. In *Proceedings of the 8th ACM Conference on Recommender Systems (RecSys '14)*. ACM, New York, NY, USA, 177–184. https://doi.org/10.1145/2645710.2645731

[27] Maksims Volkovs, Anson Wong, Zhaoyue Cheng, Felipe Perez, Ilya Stanevich, and Yichao Lu. 2019. Robust Contextual Models for In-Session Personalization. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359558

[28] Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent Aspect Rating Analysis on Review Text Data: A Rating Regression Approach. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '10)*. ACM, New York, NY, USA, 783–792. https://doi.org/10.1145/1835804.1835903

[29] Zhe Wang, Yangbo Gao, Huan Chen, and Yan Peng. 2019. Session-based Item Recommendation with Pairwise Features. In *Proceedings of the ACM Recommender Systems Challenge 2019*. ACM. https://doi.org/10.1145/3359555.3359559

[30] www.xing.com. 2016. RecSys Challenge 2016. https://2016.recsyschallenge.com/index.html. [Online; accessed 28-Aug-2019].

[31] www.xing.com. 2017. RecSys Challenge 2017. https://2017.recsyschallenge.com/index.html. [Online; accessed 28-Aug-2019].

[32] www.yoochoose.com. 2015. RecSys Challenge 2015. https://2015.recsyschallenge.com/index.html. [Online; accessed 28-Aug-2019].